Adoption of Software Testing in Open Source Projects A Preliminary Study on 50,000 Projects

Pavneet Singh Kochhar[1], Tegawendé F. Bissyandé[2], David Lo[1], and Lingxiao Jiang[1]
[1]*Singapore Management University, Singapore*
[2]*LaBRI, University of Bordeaux, France*
kochharps.2012@phdis.smu.edu.sg, bissyand@labri.fr, davidlo@smu.edu.sg, lxjiang@smu.edu.sg

***Abstract*—In software engineering, testing is a crucial activity that is designed to ensure the quality of program code. For this activity, development teams spend substantial resources constructing test cases to thoroughly assess the correctness of software functionality. What is however the proportion of open source projects that include test cases? What kind of projects are more likely to include test cases?**

In this study, we explore 50,000 projects and investigate the correlation between the presence of test cases and various project development characteristics, including the lines of code and the size of development teams.

***Keywords*-Empirical study, Software testing, Adequacy, Test cases**

I. Introduction

Project development tasks have benefited from numerous advancements in software engineering over the decades. Despite the widespread use of various strategies and tools to ensure the production of high-quality code, most software are shipped with substantial amounts of bugs. For example, in 2006, a Mozilla developer has admitted that "everyday, almost 300 bugs appear" in the project's bug tracking system [2]. Such situations are explained by various reasons, most notably by deficiencies in software testing. Ten years ago, the National Institute of Standards and Technology had conducted a study where it was reported that inadequate software testing costs the U.S economy $59.5 billions annually, i.e., about 0.6% of its GDP [11]. This alone suggests that software testing is an important task that must be integrated in the process of project development.

Software testing assesses the functionalities of a program or system and investigates whether the tested entity produces the expected results based on a set of inputs and execution contexts. Therefore, an important aim of software testing is to detect bugs. Unfortunately, since the complexity of software is intractable, software bugs are prominent and detecting all of them is practically impossible.It is commonly accepted that performing a complete testing is often infeasible. Although a large body of research about software testing has been built [3], [4], [5], [6], software programs continue to suffer from numerous defects. However, is software testing really popular in development projects? What kind of projects are more likely to include tests? These are some of the questions whose answers could shed light into the opportunity of software testing and its impact on software evolution. Our goal in this paper is indeed to fill a research gap in the importance of software testing through a large-scale empirical evaluation.

In this study, we investigate 50,000 open source projects from the GitHub hosting site. GitHub provides various developer-friendly features which makes it an important development platform for millions of software projects, including major successes such as Linux and Ruby on Rails. The contributions of this paper are as follows:

1) We conduct a large scale study on thousands of software projects. We believe it is the largest study conducted on test cases.
2) We examine the popularity of test cases in different projects and different characteristics which can affect presence of test cases.
3) We discuss our findings and show correlations between the number of test cases and various characteristics of software projects.

The structure of this paper is as follows. We elaborate our empirical study methodology in Section II. Next, we explain the statistical analysis we performed on the data in Section III, along with a number of research questions. Section IV describes the related work. We conclude and mention future work in Section V.

II. Methodology

For our empirical study, we downloaded thousands of projects using the GitHub API and selected 50,000 projects. The projects downloaded from GitHub do not have a distinct ordering scheme which varies with every request. To ensure that most of the projects are non toy projects in our dataset, we examined the dataset and found famous projects like jQuery and Ruby on Rails framework. We count the number of lines of code (LOC) in each project. We observe that over 38% of our projects have more than 500 LOC. Around 28% of the projects include more than 1,000 LOC, while more than 12% of the projects have more than 5,000 lines of code. Also, over 1,000 projects have more than 100,000 LOC which shows that our dataset includes projects of substantial sizes.

A. Collecting the dataset

a) Lines of code: We cloned the actual git repositories of the projects in the dataset and counted the number of lines of code (LOC) in each project. Since GitHub uses

the git software configuration management system (SCM) to store software revisions, we used the latest revision of the project, ignoring code comments and blank lines and used the SLOCCount[1] utility to compute the actual lines of code.

b) Test Cases: Test cases define a sequence of steps to test the behavior of functionality of an application. For each project, we collect the files whose name contains "test" and count the number of such files. Then, we examine the different characteristics of the projects with respect to the number of test cases.

c) Developer contributions: We find the contributions of developers for all the projects in our dataset i.e., we find the number of developers for each project. Git records stores the contributors name and email address for each revision of the project. Git SCM can help distinguish between revision authors, who are end contributors of code and committers, who have access to main repository and who forward the code contributions from revision authors.

B. Research questions

We investigate in this work a number of research questions to explore and assess the prevalence of software testing in software development projects. Given the extent of our datasets and the variety of artifacts that software development projects produce, we investigate various dimensions to characterize the relationship between a project and the test cases produced in that project. We are thus interested in analyzing the following research questions:

RQ1:*How many projects have test cases?* For this research question, we examine the popularity of test cases in open source projects. We also investigate the characteristics of projects containing test cases and their corresponding lines of code (LOC).

RQ2: *Does the number of developers affect the number of test cases present in a project?* Here, we wish to study the relationship between number of developers and number of test cases in a project.

III. Empirical Evaluation

In this section, we examine the research questions and report the results of our empirical study.

A. RQ1: Popularity of Test Cases

In the first research question we investigate the distribution of test cases across projects. Since we are investigating open source projects, different people can contribute either by creating the test cases or by running them and reporting bugs. Table 1 details the distribution of test cases.

We collected 50,000 projects for our experiment out of which 28,672 projects do not contain test cases. This represents 57.34% of the total projects. Remaining 42.66% of the projects contain one or more test cases.

[1]http://dwheeler.com/sloccount

Table I
Test Cases Distribution

Projects	# of Projects	% of Projects
Without Test Cases	28,672	57.34%
With Test Cases	21,328	42.66%

Table II details the prevalence of test cases: 90% of the projects have less than 100 test cases, while only 2.57% of the projects have more than 500 test cases. More than 7% of the projects have between 100 and 500 test cases, while only 17 projects have more than 10,000 test cases.

Table II
Prevalence of Test Cases

# of Test Cases	# of Projects	% of Projects with Test Cases
1-9	12,117	56.81%
10-49	5,840	27.38%
50-99	1,262	5.91%
100-249	1,138	5.33%
250-499	417	1.95%
500-999	300	1.40%
1000-4999	211	0.98%
5000-9999	26	0.12%
$>$ 10000	17	0.07%

We examine how presence/absence of test cases correlate with the lines of code. We wish to examine whether an increase in the number of LOC leads to an increase in the number of test cases. We believe that with increasing code sizes we need more test cases to sufficiently test the code and to produce reliable software.

Figure 1. Test Cases and Lines of Code

Figure 1 shows distribution of LOCs for projects with/without test cases. On average, projects without test cases have 1,548 LOC (median=244), while projects with test cases have 59,893 LOC (median=701). We performed Mann-Whitney-Wilcoxon (MWW) test to compare the set of LOC of projects with test cases and those without test cases and found that the difference between these two sets is statistically significant with p-value $< 2.2\ e^{-16}$ [2]. Thus,

[2]Here, lines of code is the dependent variable whereas project with/without test cases is an independent variable. Null hypothesis states that there is no difference in the size of projects that have test cases and the projects without test cases. Alternative hypothesis is that projects with test cases are bigger in size than projects without test cases. Given significance level α=0.001, if p-value$<\alpha$ the test rejects null hypothesis.

we can conclude that projects without test cases are smaller than projects with test cases.

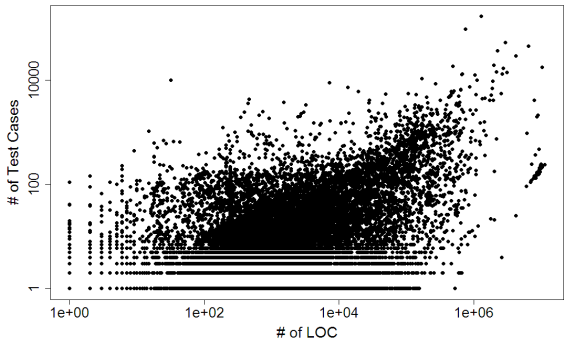

Figure 2. Correlation between Test Cases and Lines of Code

Figure 2 depicts the correlation between the number of lines of code and the number of test cases. We can observe from the graph that there is positive relationship between LOC and number of test cases. We also compute Spearman's rho which gave value 0.335 and p-value $< 2.2\ e^{-16}$ [3], suggesting that there is weak correlation between the number of test cases and the number of lines of code.

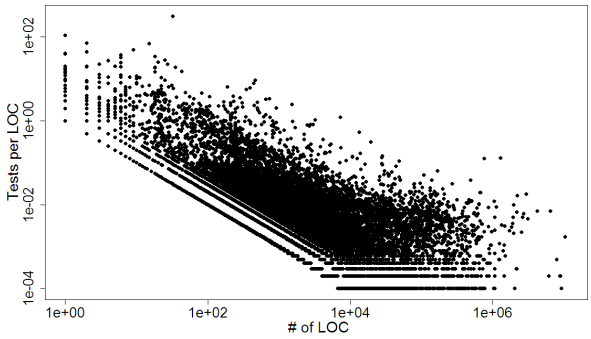

Figure 3. Correlation between Test Cases per LOC and Lines of Code

We further examine the correlation between the number of lines of code and the number of test cases per LOC. We only consider the projects with test cases. For each project, we divide the number of test cases by the number of LOC. We observe an interesting thing from Figure 3 that as the project size increases, i.e., increase in number of lines of code (LOC), we see a decrease in the number of tests per LOC. Spearman's rho for the distribution is -0.686 with p-value $< 2.2\ e^{-16}$, which shows that there is a negative correlation between the lines of code and the number of test cases per LOC.

90% of the projects have less than 100 test cases. Projects with test cases are bigger in size than projects without test cases. For the projects with test cases, the number of test cases per LOC decreases with increasing LOC.

[3]Null hypothesis (rho is zero) is rejected

B. RQ2: Developers and Test Cases

We discuss the relationship between the numbers of developers and the numbers of test cases for the projects in our dataset. Based on developer names, we have identified 2,967,146 developers who have contributed to the code bases of the projects in our dataset of 50,000 projects. In total, for projects with test cases (21,328 projects), we have identified 2,864,179 developers who may/may not have contributed test cases. On the other hand, projects without test cases have 102,967 developers and constitutes 3.4% of the total number of developers. We can observe from the statistics that projects with test cases have higher numbers of developers. Figure 4 shows that projects with/without test cases have approximately equal median number of developers. Using MWW test between the set of numbers of developers of projects with test cases and those for projects without test cases, we found p-value $< 2.2e^{-16}$ e^{-16} [4] which shows that difference between these two sets is statistically significant.

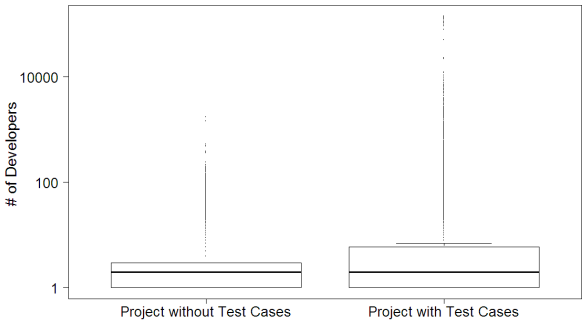

Figure 4. Number of Developers in Projects with/without Test Cases

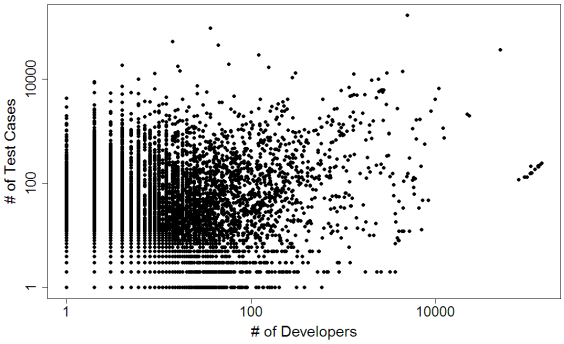

Figure 5. Test Cases and Number of Developers

Further, we investigate whether the number of developers has an effect on the presence of test cases. Figure 5 depicts the scatter plot of the numbers of developers and the numbers of test cases from the projects in our dataset. The

[4]Here, number of developers is a dependent variable whereas project with/without test cases is an independent variable. Null hypothesis states that there is no difference in the number of developers between projects without test cases and the projects with test cases. Alternative hypothesis is that projects with test cases have more number of developers. Given significance level α=0.001, if p-value$< \alpha$ the test rejects null hypothesis.

Spearman's rho for the distribution is 0.207 with p-value $<$ 2.2 e^{-16} suggesting that there is weak correlation between numbers of developers and test cases.

We proceed to investigate correlation between the numbers of test cases per developer and the numbers of developers in the project. Figure 6 shows the scatter plot between the two data variables. Here, we consider only the projects with non-zero test cases. We observe a decrease in the number of test cases per developer with an increase in the number of developers. The Spearman's rho for the two distributions is -0.446 (p-value $<$ 2.2 e^{-16}). We observe that the relationship between the number of developers and the number of test cases per developer is negative. Thus, we can conclude that test count per developer decreases with an increase in the number of developers as only some of the developers contribute the test cases.

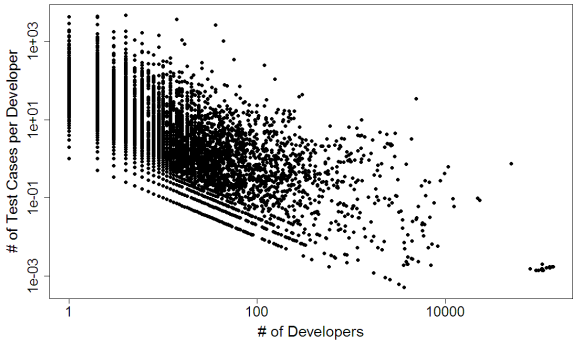

Figure 6. Correlation between # of Test Cases per Developer and # of Developers

Although the number of test cases increases with an increase in the number of developers, number of test cases per developer decreases with this increase.

Threats to Validity. We use SLOCCount[1] utility to count the number of lines of code. This tool covers 27 programming languages. It is possible that we miss some lines of code that are not written in one of these languages. We use heuristics to detect test files, i.e., we consider the files whose name contains test. This might not identify all test files whose name does not have word test and conversely, detect some files whose name contain the word test but actually are not test files. In order to scale to 50,000 projects, we need to take these heuristics.

IV. Related Work

There is a large research community, which uses empirical methods to study several aspects of software engineering. Ajila and Wu perform an empirical study on the adoption of open source software development [1]. Glynn et al. investigate the adoption of open source software in commercial applications [7]. Greiler et al. conduct a qualitative study of test practices followed by a community of people working on plug-in based applications [8]. Previously, there have been several large scale studies on software projects. Gruska et al. evaluates a lightweight anomaly detection technique on a collection of 6,000 projects [9]. Surian et al. analyze a snapshot of projects in SourceForge.Net [10]. In this work, we conduct a large scale empirical study using 50,000 projects from a developer-oriented platform, namely GitHub.

V. Conclusion and Future Work

Test cases are important artefacts in any software project which allow developers test their code and produce reliable software. We conduct a large scale study to examine the presence/absence of test cases with respect to other characteristics of a project such as lines of code and number of developers. We plot graphs between test cases and these characteristics and use statistical analysis to examine the correlation between them. Our analysis shows the following results

1) Projects having test cases are bigger in size than projects without test cases. However, as projects get larger the number of tests per LOC decreases.
2) Projects having bigger team size have higher number of test cases whereas the number of test cases per developer decreases with an increase in the size of the development team.

In this paper, we have limited our study to projects collected in June 2012. In future, we can expand our studies and include more projects and consider other characteristics such as number of bugs and programming languages which can affect presence of test cases.

References

[1] S. Ajila and D. Wu, "Empirical study of the effects of open source adoption on software development economics," *Journal of Systems and Software*, 2007.
[2] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in *ICSE*, 2006.
[3] V. Basili and R. Selby, "Comparing the effectiveness of software testing strategies," *Software Engineering, IEEE Transactions on*, 1987.
[4] L. Briand and Y. Labiche, "Empirical studies of software testing techniques: Challenges, practical strategies, and future research," *ACM SIGSOFT Software Engineering Notes*, 2004.
[5] P. Frankl and O. Iakounenko, "Further empirical studies of test effectiveness," in *ACM SIGSOFT Software Engineering Notes*, 1998.
[6] P. Frankl and S. Weiss, "An experimental comparison of the effectiveness of branch testing and data flow testing," *Software Engineering, IEEE Transactions on*, 1993.
[7] E. Glynn, B. Fitzgerald, and C. Exton, "Commercial adoption of open source software: an empirical study," in *ISESE*, 2005.
[8] M. Greiler, A. van Deursen, and M.-A. D. Storey, "Test confessions: A study of testing practices for plug-in systems," in *ICSE*, 2012.
[9] N. Gruska, A. Wasylkowski, and A. Zeller, "Learning from 6, 000 projects: lightweight cross-project anomaly detection," in *ISSTA*, 2010.
[10] D. Surian, D. Lo, and E.-P. Lim, "Mining collaboration patterns from a large developer network," in *WCRE*, 2010.
[11] G. Tassey, "The economic impacts of inadequate infrastructure for software testing," *National Institute of Standards and Technology, RTI Project*, 2002.